

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION  
International Bureau



INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification <sup>6</sup> :	H04L 12/00	A2	(11) International Publication Number: WO 99/44334
(21) International Application Number:	PCT/US99/04069		(43) International Publication Date: 2 September 1999 (02.09.99)
(22) International Filing Date:	25 February 1999 (25.02.99)		
(30) Priority Data:	60/076,048 26 February 1998 (26.02.98) US 09/044,934 20 March 1998 (20.03.98) US		(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).
(71) Applicant:	SUN MICROSYSTEMS, INC. [US/US]; 901 San Antonio Road, MS UPAL01-521, Palo Alto, CA 94303 (US).		<b>Published</b> <i>Without international search report and to be republished upon receipt of that report.</i>
(72) Inventors:	ARNOLD, Kenneth, C., R.; 7 Moon Hill Road, Lexington, MA 02173 (US). WALDO, James, H.; 155 Ruby Road, Dracut, MA 01826 (US).		
(74) Agents:	GARRETT, Arthur, S.; Finnegan, Henderson, Farra bow Garrett & Dunner, L.L.P., 1300 I Street, N.W., Washington, DC 20005-3315 (US) et al.		

(54) Title: POLYMORPHIC TOKEN BASED CONTROL

(57) Abstract

The protocol for controlling a network is encapsulated within the token circulated through the token ring network. Each computer in the network that receives the token examines the token and implements the network protocol specified in the token. In this manner, the protocol of the network can be easily changed, and automatically promulgated throughout the network.

**FOR THE PURPOSES OF INFORMATION ONLY**

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece	ML	Mali	TR	Turkey
BG	Bulgaria	HU	Hungary	MN	Mongolia	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MR	Mauritania	UA	Ukraine
BR	Brazil	IL	Israel	MW	Malawi	UG	Uganda
BY	Belarus	IS	Iceland	MX	Mexico	US	United States of America
CA	Canada	IT	Italy	NE	Niger	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NL	Netherlands	VN	Viet Nam
CG	Congo	KE	Kenya	NO	Norway	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NZ	New Zealand	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	PL	Poland		
CM	Cameroon	KR	Republic of Korea	PT	Portugal		
CN	China	KZ	Kazakhstan	RO	Romania		
CU	Cuba	LC	Saint Lucia	RU	Russian Federation		
CZ	Czech Republic	LI	Liechtenstein	SD	Sudan		
DE	Germany	LK	Sri Lanka	SE	Sweden		
DK	Denmark	LR	Liberia	SG	Singapore		
EE	Estonia						

## POLYMORPHIC TOKEN BASED CONTROL

Related Applications

The following identified U.S. patent applications are relied upon and are incorporated by reference in this application.

5        Provisional U.S. Patent Application No. 60/076,048, entitled "Distributed Computing System," filed on February 26, 1998.

U.S. Patent Application No. 09/044,923, entitled "Method and System for Leasing Storage," bearing attorney docket no. 06502.0011-01000, and filed on the same date herewith.

10      U.S. Patent Application No. 09/044,838, entitled "Method, Apparatus, and Product for Leasing of Delegation Certificates in a Distributed System," bearing attorney docket no. 06502.0011-02000, and filed on the same date herewith.

U.S. Patent Application No. 09/044,834, entitled "Method, Apparatus and Product for Leasing of Group Membership in a Distributed System," bearing attorney docket no. 06502.0011-03000, and filed on the same date herewith.

15      U.S. Patent Application No. 09/044,916, entitled "Leasing for Failure Detection," bearing attorney docket no. 06502.0011-04000, and filed on the same date herewith.

U.S. Patent Application No. 09/044,933, entitled "Method for Transporting Behavior in Event Based System," bearing attorney docket no. 06502.0054-00000, and filed on the same date herewith.

20      U.S. Patent Application No. 09/044,919, entitled "Deferred Reconstruction of Objects and Remote Loading for Event Notification in a Distributed System," bearing attorney docket no. 06502.0062-01000, and filed on the same date herewith.

U.S. Patent Application No. 09/044,938, entitled "Methods and Apparatus for Remote Method Invocation," bearing attorney docket no. 06502.0102-00000, and filed on the same date herewith.

25      U.S. Patent Application No. 09/045,652, entitled "Method and System for Deterministic Hashes to Identify Remote Methods," bearing attorney docket no. 06502.0103-00000, and filed on the same date herewith.

U.S. Patent Application No. 09/044,790, entitled "Method and Apparatus for Determining Status of Remote Objects in a Distributed System," bearing attorney docket no. 06502.0104-00000, and filed on the same date herewith.

5 U.S. Patent Application No. 09/044,930, entitled "Downloadable Smart Proxies for Performing Processing Associated with a Remote Procedure Call in a Distributed System," bearing attorney docket no. 06502.0105-00000, and filed on the same date herewith.

10 U.S. Patent Application No. 09/044,917, entitled "Suspension and Continuation of Remote Methods," bearing attorney docket no. 06502.0106-00000, and filed on the same date herewith.

15 U.S. Patent Application No. 09/044,835, entitled "Method and System for Multi-Entry and Multi-Template Matching in a Database," bearing attorney docket no. 06502.0107-00000, and filed on the same date herewith.

20 U.S. Patent Application No. 09/044,839, entitled "Method and System for In-Place Modifications in a Database," bearing attorney docket no. 06502.0108, and filed on the same date herewith.

25 U.S. Patent Application No. 09/044,945, entitled "Method and System for Typesafe Attribute Matching in a Database," bearing attorney docket no. 06502.0109-00000, and filed on the same date herewith.

30 U.S. Patent Application No. 09/044,931, entitled "Dynamic Lookup Service in a Distributed System," bearing attorney docket no. 06502.0110-00000, and filed on the same date herewith.

U.S. Patent Application No. 09/044,939, entitled "Apparatus and Method for Providing Downloadable Code for Use in Communicating with a Device in a Distributed System," bearing attorney docket no. 06502.0112-00000, and filed on the same date herewith.

25 U.S. Patent Application No. 09/044,826, entitled "Method and System for Facilitating Access to a Lookup Service," bearing attorney docket no. 06502.0113-00000, and filed on the same date herewith.

30 U.S. Patent Application No. 09/044,932, entitled "Apparatus and Method for Dynamically Verifying Information in a Distributed System," bearing attorney docket no. 06502.0114-00000, and filed on the same date herewith.

U.S. Patent Application No. 09/030,840, entitled "Method and Apparatus for Dynamic Distributed Computing Over a Network," and filed on February 26, 1998.

U.S. Patent Application No. 09/044,936, entitled "An Interactive Design Tool for Persistent Shared Memory Spaces," bearing attorney docket no. 06502.0116-00000, and filed on the same date herewith.  
5

U.S. Patent Application No. 09/044,915, entitled "Stack-Based Access Control," bearing attorney docket no. 06502.0118-00000, and filed on the same date herewith.

U.S. Patent Application No. 09/044,944, entitled "Stack-Based Security Requirements," bearing attorney docket no. 06502.0119-00000, and filed on the same date herewith.  
10

U.S. Patent Application No. 09/044,837, entitled "Per-Method Designation of Security Requirements," bearing attorney docket no. 06502.0120-00000, and filed on the same date herewith.

#### Field of the Invention

This invention relates generally to local area networks and, more specifically, to token passing in a token ring local area network.  
15

#### Background of the Invention

Computers in a computer network often share a limited number of resources. One conventional method of allocating access between shared resources involves passing a "token" circularly to each computer in the network. The computers agree ahead of time that when using this token protocol, only the computer that has possession of the token may access the resource. A popular example of a network using a token passing algorithm is a token ring network.  
20

Token ring networks are baseband networks, which means that all the transmission capacity (i.e., network bandwidth) of the network media is used by one signal. Because only one signal at a time can be transmitted over the network, multiple computers in a token ring network must not transmit simultaneously. This is accomplished using a token access protocol.  
25

In the token access protocol, computers in the network agree to continuously circulate an information frame to all the computers in the network. When a computer wants to send a message, it waits until it possesses the empty frame, and then modifies the frame by inserting: its message, a destination identifier, and a "token." The token may simply be, for example, a bit field in the frame that the inserting computer changes to a 1 to indicate a token is present or a 0 to indicate an empty frame.

The frame is examined by each computer as it is passed around the network. The destination computer copies the message from the frame and changes the token back to zero. The originating computer, when it receives the frame, can verify that its message was received by noticing that the token has been set to zero. The originator then removes the message from the frame and passes the empty frame to the next computer in the network.

Although conventional token ring networks are effective at preventing data collisions, they have disadvantages. In particular, in order to implement a token ring network, all the computers in the network must agree ahead of time on the appropriate protocol to use in passing the message frame. This can be difficult, if, for example, the network administrator wishes to change the protocol of the token ring, as each computer must be updated before the network is operational. It is therefore desirable to improve token ring networks.

#### Summary of the Invention

Objects and advantages of the invention will be set forth in part in the description which follows, and in part will be obvious from the description, or may be learned by practice of the invention. The objects and advantages of the invention will be realized and attained by means of the elements and combinations particularly pointed out in the appended claims.

To achieve the objects and in accordance with the purpose of the invention, as embodied and broadly described herein, a first aspect consistent with the present invention includes a method of updating a protocol for controlling a computer network including a plurality of computers, the method comprises the steps of: (1) creating a token object containing methods defining an updated version of the protocol; (2) sequentially passing the token object to each computer in the network; and (3) updating the protocol used by each of the plurality of

computers with the methods defining the updated version of the protocol while the token object is present at each computer.

A second aspect consistent with the present invention is directed to a token ring network. The network comprises a plurality of computers coupled together and a token ring object. The token ring object includes methods and data that define a protocol for the token ring network, the token ring object is sequentially transferred to each of the plurality of computers, and when one of the plurality of computers has received possession of the token ring object, it adopts the protocol defined by the token ring object when the protocol defined by the token ring object is different than the protocol in use by the computer.

Further, a third aspect consistent with the present invention includes a method of updating a protocol for controlling a computer network. The method includes the steps of: (1) receiving a token object at a first computer in the network; (2) consulting the token object, and when the token object indicates that a new protocol is to be used to transmit information on the network, updating an older version of the protocol stored at the first computer; and (3) transmitting the token object to a second computer in the network, the second computer being determined based on information in the token object.

Further, an additional aspect consistent with the present invention includes a method of updating a protocol for controlling a computer network. The method includes the steps of: (1) receiving a token object defining a protocol of the network; and (2) sending the object using the protocol defined in the token object.

Still further, an additional aspect consistent with the present invention includes a computer readable memory device containing token including an indication of a protocol to be used when communicating in a network.

25

#### Brief Description of the Drawings

The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate several embodiments consistent with this invention and, together with the description, help explain the principles of the invention. In the drawings,

Fig. 1 is a high-level diagram of a token ring network;

The JavaSpace 222 is an object repository used by programs within the exemplary distributed system 100 to store objects. Programs use the JavaSpace 222 to store objects persistently as well as to make them accessible to other devices within the exemplary distributed system. Java spaces are described in greater detail in co-pending U.S. Patent Application No. 5 08/971,529, entitled "Database System Employing Polymorphic Entry and Entry Matching," assigned to a common assignee, filed on November 17, 1997, which is incorporated herein by reference. One skilled in the art will appreciate that the exemplary distributed system 100 may contain many lookup services, discovery servers, and JavaSpaces.

Although systems and methods consistent with the present invention are described as operating in the exemplary distributed system and the Java programming environment, one skilled in the art will appreciate that the present invention can be practiced in other systems and other programming environments. Additionally, although aspects of the present invention are described as being stored in memory, one skilled in the art will appreciate that these aspects can also be stored on or read from other types of computer-readable media, such as secondary storage devices, like hard disks, floppy disks, or CD-ROM; a carrier wave from the Internet; or other forms of RAM or ROM. Sun, Sun Microsystems, the SunLogo, Java, and Java-based trademarks are trademarks or registered trademarks of Sun Microsystems Inc. in the United States and other countries.

#### *Polymorphic Token Passing*

A token ring network consistent with the present invention passes a polymorphic token object around the network in place of the static token frame used in conventional token ring networks. The passing of the token object is preferably implemented using a distributed object-oriented programming environment, such as Java RMI (described above). Java RMI is especially suitable to the present invention, as it provides for the automatic management of distributed objects and the ability to easily pass objects from machine to machine on a network.

Fig. 3 is a diagram illustrating an exemplary token 302 and its relationship with a token class hierarchy 304. Token 302 is preferably implemented using an object data structure, and as such, may include functionality (e.g., methods) and data. As used throughout this specification, and as generally used in the object-oriented programming field, a class refers to a template from which objects may be defined. An object is an instance of a particular class

Fig. 2 is a block diagram illustrating an exemplary embodiment of a computer used in the token ring network;

Fig. 3 is a diagram illustrating an embodiment of a token object consistent with the present invention; and

5 Fig. 4 is a flow chart illustrating methods consistent with the present invention.

#### Detailed Description

A token ring network is disclosed in which tokens passed between computers in the network define a protocol, or at least a portion of the protocol, for the token ring network. Each computer in the network that receives the token examines the token and implements the network protocol specified in the token. Any computer having appropriate permission may change or update the protocol in the token, and thereby change the protocol for the entire network.

Reference will now be made in detail to the embodiments of the invention, examples of which are illustrated in the accompanying drawings.

15

#### *System Overview*

Fig. 1 is a high level diagram of a token ring network 100 made up of four distributed computers 102, 104, 106, and 108 passing a token object in the counter clockwise direction through network media 120. The token object is preferably passed between computers 102-108 using some form of remote object passing mechanism, such as the Javaremove invocation system (Java RMI). Additionally, one of computers 102-108 may act as a gateway to a larger token ring network or to a non token ring network. As shown in Fig. 1, computer 106 acts as a gateway to the Internet network 110.

In exemplary distributed system 100, different computers and devices are federated into what appears to the user to be a single system. By appearing as a single system, the distributed system 100 provides the simplicity of access and the power of sharing that can be provided by a single system without giving up the flexibility and personalized response of a personal computer or workstation. Distributed system 100 may contain thousands of devices operated by users who are geographically disperse, but who agree on basic notions of trust, administration, and policy.

30

Within the distributed 100 system are various logical groupings of services provided by one or more devices, and each such logical grouping is known as a Djinn. A "service" refers to a resource, data, or functionality that can be accessed by a user, program, device, or another service and that can be computational, storage related, communication related, or related to providing access to another user. Examples of services provided as part of a Djinn include devices, such as printers, displays, and disks; software, such as applications or utilities; information, such as databases and files; and users of the system.

Both users and devices may join a Djinn. When joining a Djinn, the user or device adds zero or more services to the Djinn and may access, subject to security constraints, any one of the services it contains. Thus, devices and users federate into a Djinn to share access to its services. The services of the Djinn appear programmatically as objects of the Java programming environment, which may include other objects, software components written in different programming languages, or hardware devices. A service has an interface defining the operations that can be requested of that service, and the type of the service determines the interfaces that make up that service.

The Java RMI and its relationship with computers 102-108 and token ring network 100 will now briefly be described with reference to Fig. 2.

Fig. 2 depicts computer 102 in greater detail showing a number of the software components of the distributed system 100. Computer 102 includes a memory 202, a secondary storage device 204, a central processing unit (CPU) 206, an input device 208, and a video display 210. The memory 202 includes a lookup service 212, a discovery server 214, and a Java™ runtime system 216. The Java runtime system 216 includes the Java™ remote method invocation system (RMI) 218 and a Java™ virtual machine 220. The secondary storage device 204 includes a JavaSpace™ 222.

The exemplary distributed system 100 is based on the Java programming environment and thus makes use of the Java runtime system 216. The Java runtime system 216 includes the Java API, allowing programs running on top of the Java runtime system to access, in a platform-independent manner, various system functions, including windowing capabilities and networking capabilities of the host operating system. Since the Java API provides a single common API across all operating systems to which the Java runtime system is ported, the programs running

on top of a Java runtime system run in a platform-independent manner, regardless of the operating system or hardware configuration of the host platform. The Java runtime system 216 is provided as part of the Java software development kit available from Sun Microsystems of Mountain View, CA.

5       The Java virtual machine 220 also facilitates platform independence. The Java virtual machine 220 acts like an abstract computing machine receiving instructions from programs in the form of byte codes and interpreting these byte codes by dynamically converting them into a form for execution, such as object code, and executing them. RMI 218 facilitates remote method invocation by allowing objects executing on one computer or device to invoke methods 10 of an object on another computer or device. Both RMI and the Java virtual machine are also provided as part of the Java software development kit.

15      The lookup service 212 defines the services that are available for a particular Djinn. That is, there may be more than one Djinn and, consequently, more than one lookup service within the exemplary distributed system 100. The lookup service 212 contains one object for each service within the Djinn, and each object contains various methods that facilitate access to the corresponding service. The lookup service 212 and its access are described in greater detail in co-pending U.S. Patent Application No. \_\_\_\_\_, entitled "Method and System for Facilitating Access to a Lookup Service," which has been previously incorporated by reference.

20      The discovery server 214 detects when a new device is added to the exemplary distributed system 100, during a process known as boot and join or discovery, and when such a new device is detected the discovery server passes a reference to the lookup service 212 to the new device so that the new device may register its services with the lookup service and become a member of the Djinn. After registration, the new device becomes a member of the Djinn, and as a result, it may access all the services contained in the lookup service 212. The process of 25 boot and join is described in greater detail in co-pending U.S. Patent Application No. \_\_\_\_\_, entitled "Apparatus and Method for providing Downloadable Code for Use in Communicating with a Device in a Distributed System," which has previously been incorporated by reference.

and can include attribute information that distinguishes objects of the same class. Objects inherit behavior from the class they depend from. Token object 302, for example, is an instance of, and inherits behavior from, "secure token" class 305, which in turn inherits behavior from "general token" class 303.

As shown, token object 302 includes method(s) 310 defining the token passing order in the network (e.g., counter clock-wise), method(s) 311 defining a distress protocol to be used by a malfunctioning computer, method(s) 312 defining network diagnostic checking routines, and method(s) 313 defining security measures to be implemented by the network. In the context of conventional token ring networks, token ring protocols that implement the functionality defined by methods 310-313 are well known, and accordingly, a detailed description of these methods is omitted.

Token 302 also includes a message data field 314, a destination data field 315, and a token data field 316, each of which is directly analogous to the message, message identifier, and token described above regarding the conventional token ring network frame.

Token class 303 defines the general functionality required by a "token." Classes and objects defined from the general class 303 inherit this functionality. As shown, token class 303 implements, or partially implements, methods 310-312 and fields 314-316. Secure token class 305 is a subclass of class 303, and as such, class 305 inherits the functionality of class 303. Additionally, subclass 305 may define its own methods and variables, including, for example, method(s) 313 defining network security measures. Quick token 306 is also a subclass of class 303. Quick token 303 may include, for example, method(s) 317 that further define the passing order defined in method(s) 310.

In operation, each computer in network 100 examines the token object it receives and, based on this examination, modifies the protocol it uses to implement the token ring network. If a computer wishes to change the token network protocol of the network, the computer simply changes methods in the token object by either updating, overriding, or adding a new method. As the token object propagates through the network, the new protocol is implemented.

Fig. 4 is a flow chart illustrating methods consistent with the present invention. Preferably, to ensure network integrity, only authorized computers should be able to modify the token ring network protocol. If a computer wishes to modify the protocol, (step 402), and it has

appropriate authority, (step 403), it modifies the protocol simply by adding or substituting, when it has control of the token object, its new methods that define the token ring protocol (step 404). Whether a computer has authorization to modify the network protocol may be indicated by, for example, a field in the token, or pre-hardwired into the computers in network 100.

5        Each computer in network 100 that receives the token object consults the token object and, if necessary, updates its version of the network protocol (step 405). The computer may then appropriately operate on the token and pass the token to the next computer in the network (step 406).

10      As an illustration of the method shown in Fig. 4, assume a computer would like to change the present token object, which is an instance of secure token class 305, to a quick token object, which is an instance of class 306. Assuming the computer was authorized to change the token object, it would wait until it receives the secure token object, substitute the secure token object with the quick token object, consult the quick token object for the appropriate protocol, and then pass the quick token object to the next computer in the network.

15      While there has been illustrated and described what are at present considered to be preferred embodiments and methods of the present invention, it will be understood by those skilled in the art that various changes and modifications may be made, and equivalents may be substituted for elements thereof without departing from the true scope of the invention. For example, while the foregoing systems and methods have been described with reference to a Java-based, runtime environment, other run-time environments could conceivably be used to implement the present invention. Further, although the above-discussed embodiment was discussed in the context of a token ring network, one of ordinary skill in the art will appreciate that token objects consistent with the present invention could be applied equally as well to any token passing algorithm used by a network.

20      In addition, many modifications may be made to adapt a particular element, technique or implementation to the teachings of the present invention without departing from the central scope of the invention. Therefore, it is intended that this invention not be limited to the particular embodiments and methods disclosed herein, but that the invention include all embodiments falling within the scope of the appended claims.

What is Claimed:

1. A method of updating a resource allocation protocol for controlling a computer network including a plurality of computers, the method comprising the steps of:  
creating a token object containing a method defining an updated version of the protocol;  
sequentially passing the token object to each computer in the network to facilitate communication between the computers; and

5 updating the protocol used by each of said plurality of computers with the method defining the updated version of the protocol while the token object is present at one of the computers.

10 2. The method of claim 1, wherein the step of sequentially passing the token object to each computer in the network includes the substep of introducing the token object into the network by an authorized computer when the authorized computer is given control of a previous version of the token object.

15 3. The method of claim 1, wherein the step of sequentially passing the token object includes the substep of transmitting a token object defined with the Java programming language.

4. The method of claim 3, wherein the step of sequentially passing the token object further includes the substep of transmitting the token object using the Java remote invocation system.

20 5. The method of claim 1, wherein the protocol carried out by the plurality of computers in the network implements a token ring computer network.

25 6. The method of claim 1, wherein the step of creating the token object containing methods defining an updated version of the protocol further includes the substep of defining a new token passing order for the network.

12. The method of claim 10, wherein the step of transmitting the token object includes the substep of transmitting the token object using the Java remote invocation system.

5           13. The method of claim 11, wherein the step of introducing an updated version of the token object further includes the substep of defining a new token transmission order for the network.

10           14. A computer readable medium containing instructions for causing computers to update a protocol used to control a computer network, the instructions causing the computers to perform the steps of:

creating a token object containing methods defining an updated version of the protocol; sequentially passing the token object to each of the computers; and

updating the protocol used by each of the computers with the methods defining the updated version of the protocol while the token object is present at said each computer.

15           15. The computer readable medium of claim 14, wherein the instructions for causing the computers to perform the step of sequentially passing the token object to each of the computers includes the substep of introducing the token object to the computers by an authorized computer when the authorized computer is given control of a previous version of the token object.

20           16. The computer readable medium of claim 14, wherein the instructions for causing the computers to perform the step of sequentially passing the token object includes the substep of transmitting the token object using the Java remote invocation system.

25           17. The computer readable medium of claim 14, wherein the protocol carried out by the plurality of computers in the network implements a token ring computer network.

7. A token ring network comprising:

a token ring object including methods and data that define a protocol for the token ring network,

5 a plurality of interconnected computers; and

the token ring object being sequentially transferred to each of the plurality of computers to facilitate communication between the plurality of computers, and when one of said plurality of computers has received possession of the token ring object, adopting the protocol defined by the token ring object when the protocol defined by the token ring object is different than the protocol in use by the computer.

10

8. The network of claim 7, wherein each of said plurality of computers further includes a remote object passing mechanism to transfer the token object to other of the plurality of computers.

15

9. The network of claim 8, wherein each of said plurality of computers further includes a virtual machine on which the remote object passing mechanism is implemented.

10. A method of updating a protocol for controlling a computer network comprising the steps of:

20 receiving a token object at a first computer in the network;

consulting the token object, and, when the token object indicates that a new protocol is to be used to transmit information on the network, updating an older version of the protocol stored at the first computer; and

transmitting the token object to a second computer in the network, the second computer 25 being determined based on information in the token object.

11. The method of claim 10, further including the step of introducing an updated version of the token object into the network.

18. The computer readable medium of claim 14, wherein the instructions for causing the computers to perform the step of creating the token object containing methods defining an updated version of the protocol further includes the substep of defining a new token passing order for the network.

5

19. A method for updating a protocol in a network comprising the steps of:  
receiving a token object defining a protocol of the network; and  
sending the token object using the protocol defined in the token object.

10 20. The method of claim 19, wherein the receiving step includes the substep of receiving the token object at a first computer in the network and the sending step includes the substep of transmitting the object to a second computer in the network.

15 21. The method of claim 19, further including the step of sequentially sending the token object to each computer in the network.

22. The method of claim 19, further including the step of introducing an updated version of the token object into the network.

20 23. A computer readable memory device containing:  
a token including an indication of a protocol to be used when communicating in a network.

25 24. The computer readable memory device of claim 23, wherein the token is an object.

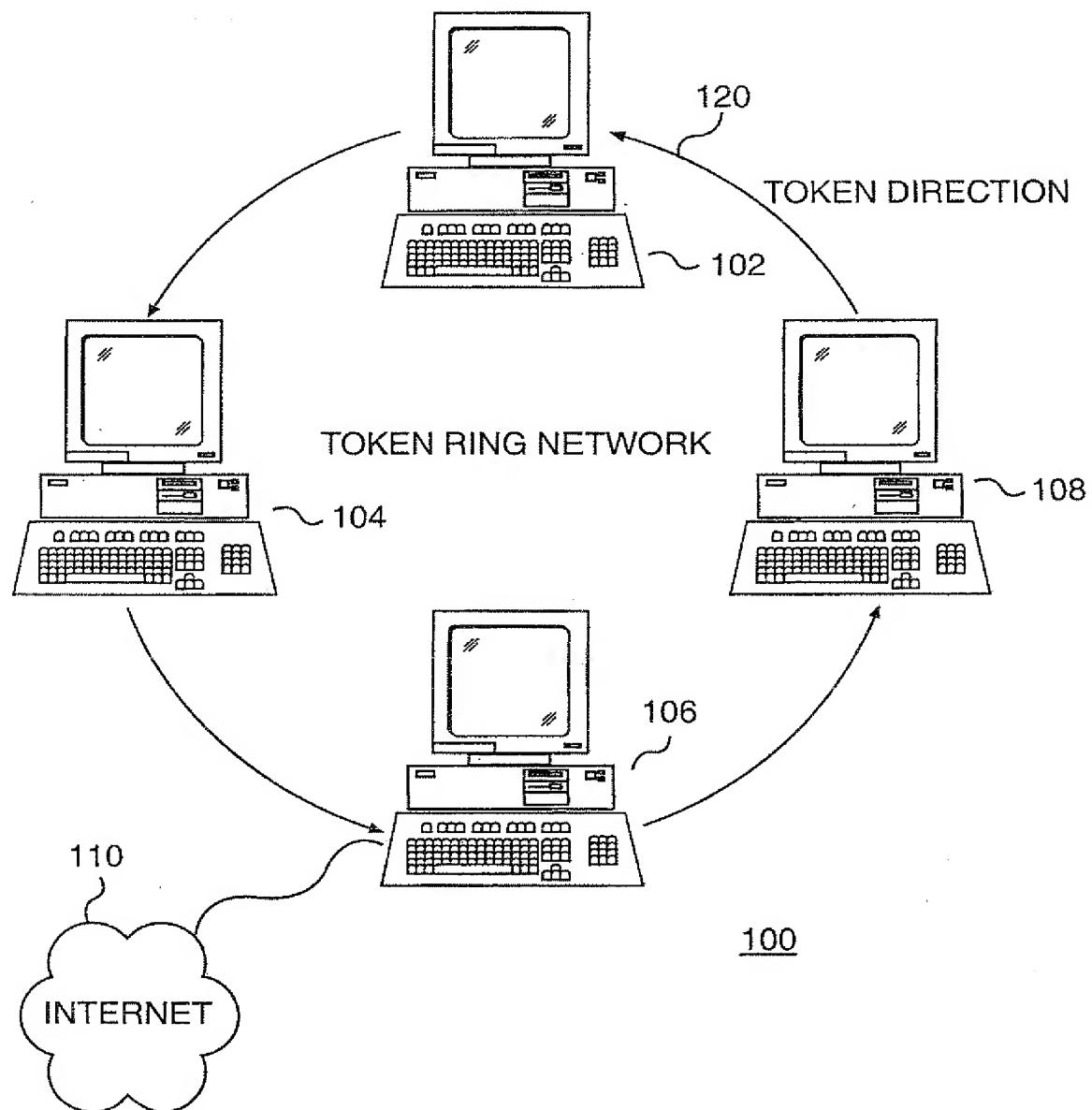
25 25. The computer readable medium of claim 23, wherein the token further includes a method for defining a token passing order in the network.

30 26. The computer readable medium of claim 23, wherein the token further includes methods defining network diagnostics.

27. The computer readable medium of claim 23, wherein the token further includes a message data field and a destination data field.

28. The computer readable medium of claim 23, wherein the token further includes  
5 methods defining security procedures for the network.

1/4

**FIG. 1**

2/4

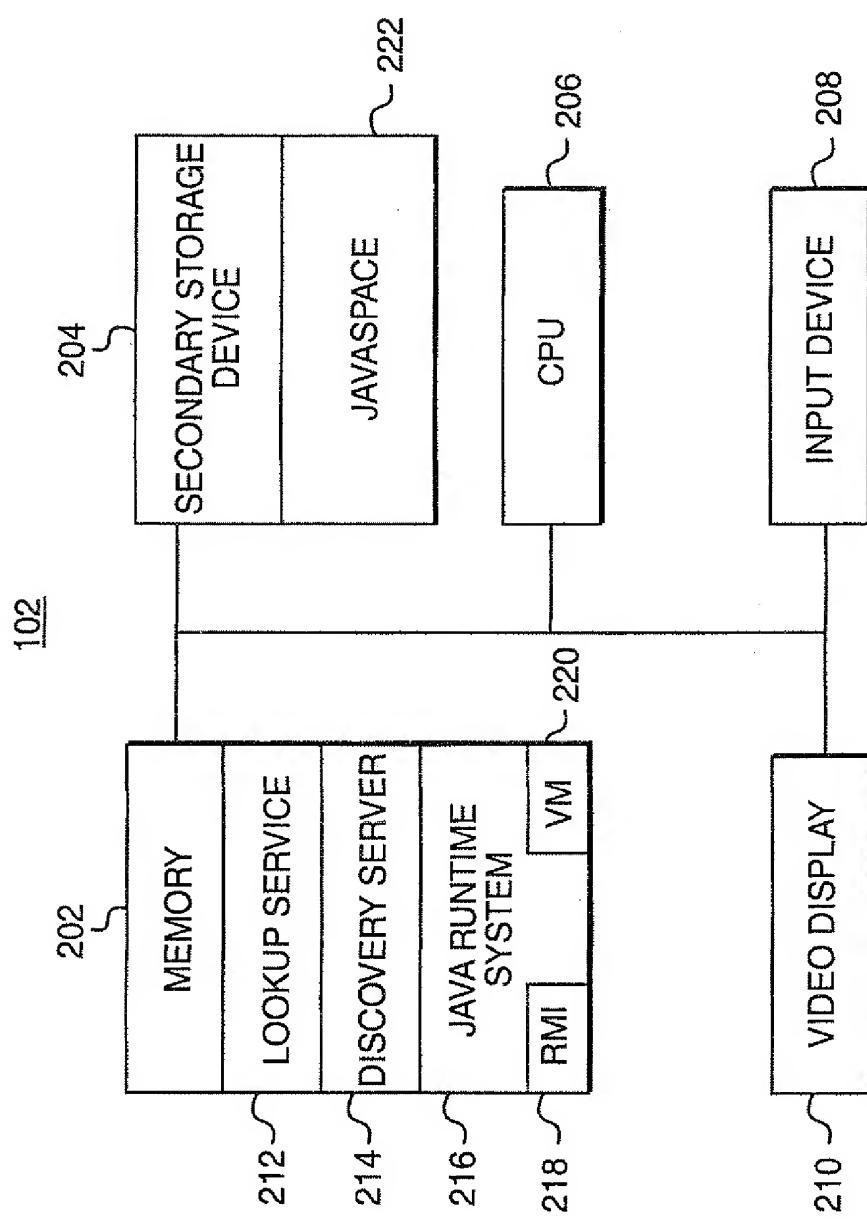
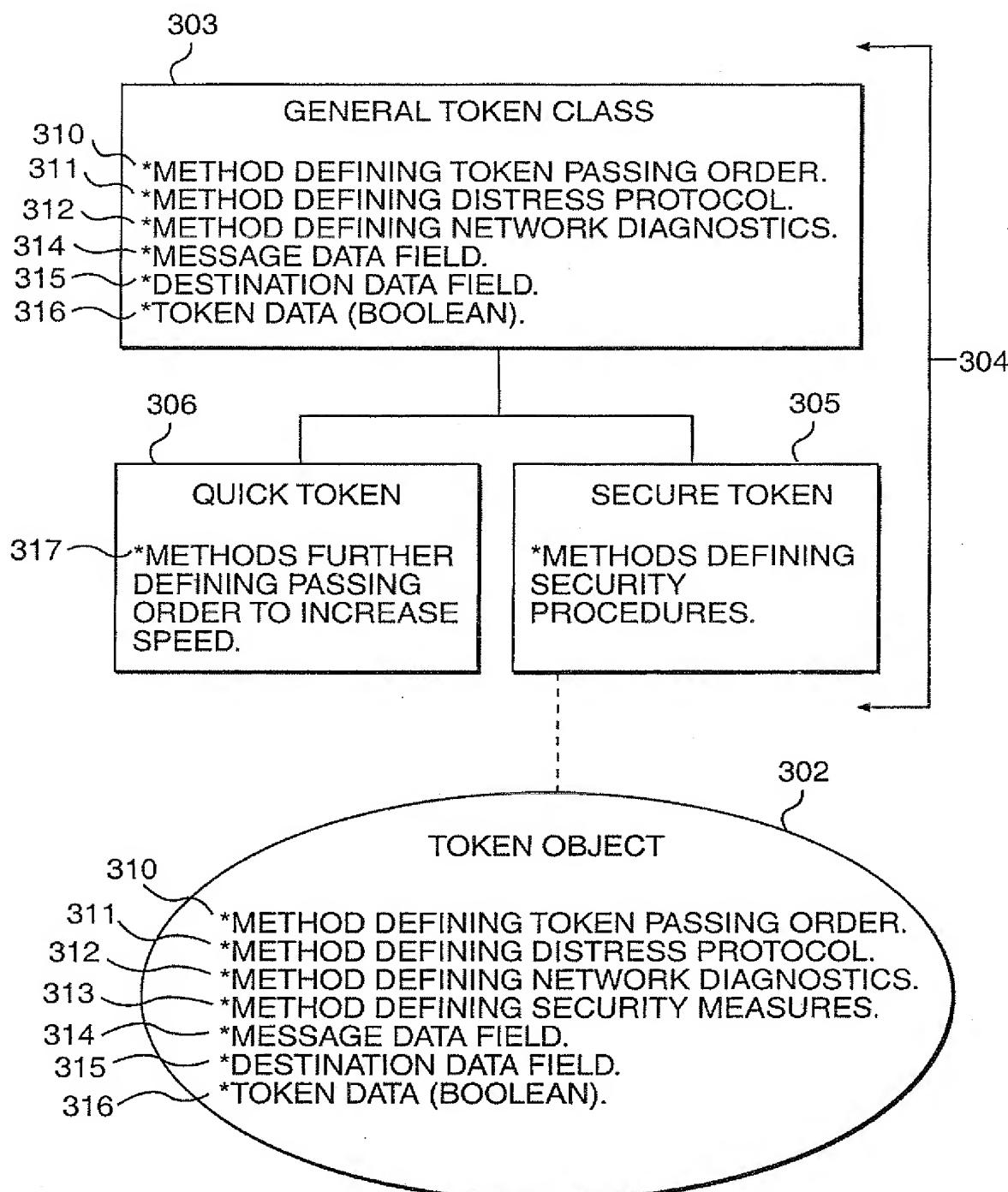
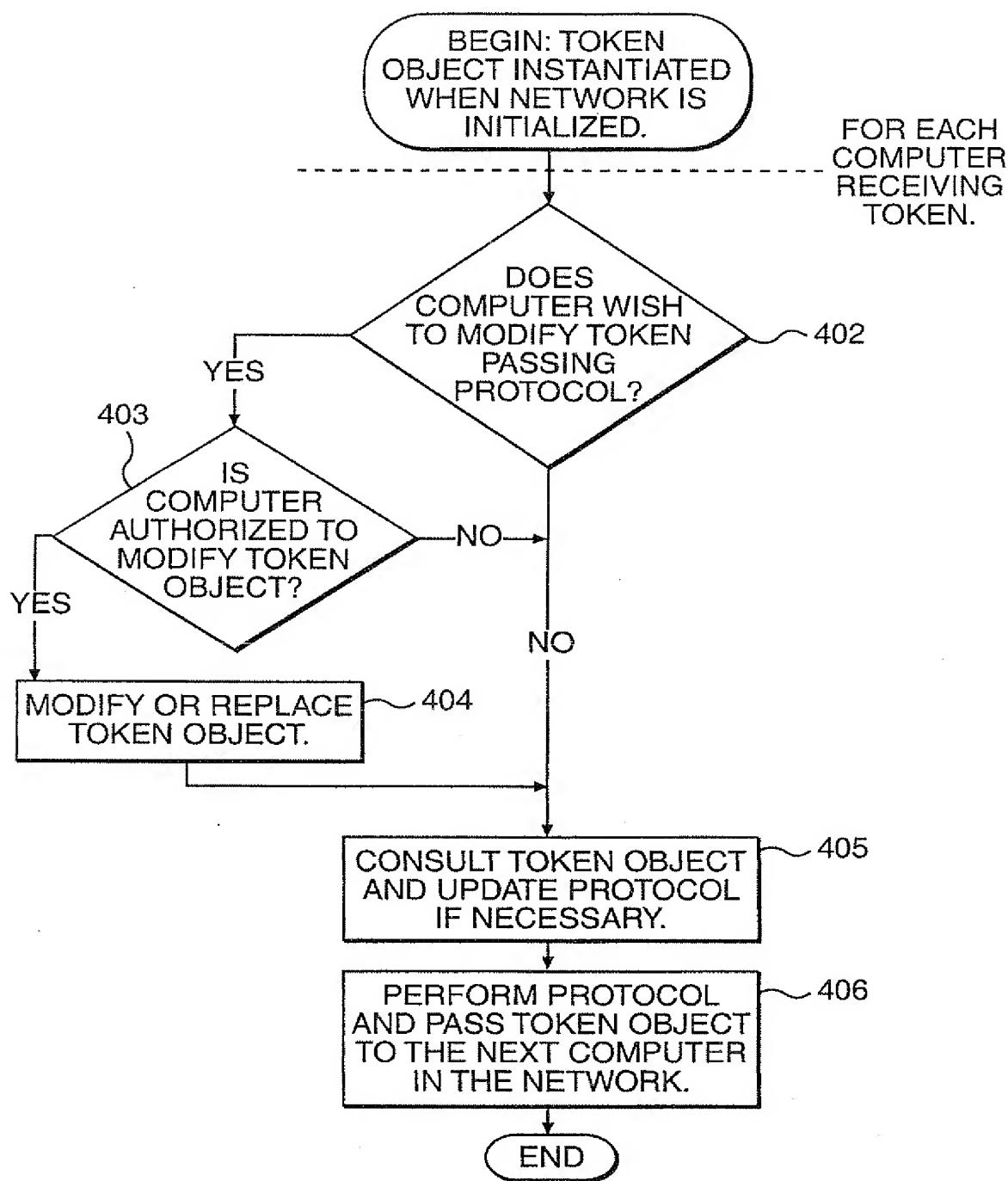


FIG. 2

3/4

**FIG. 3**

4/4

**FIG. 4**